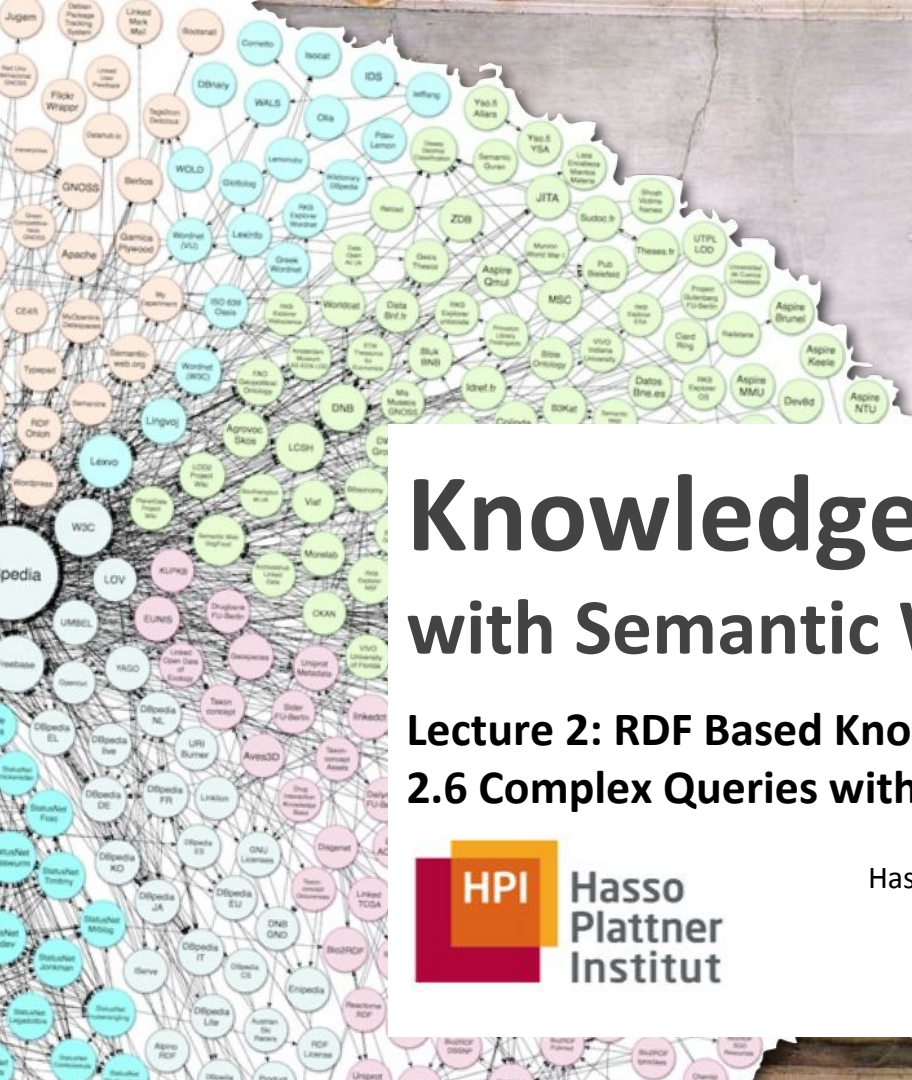


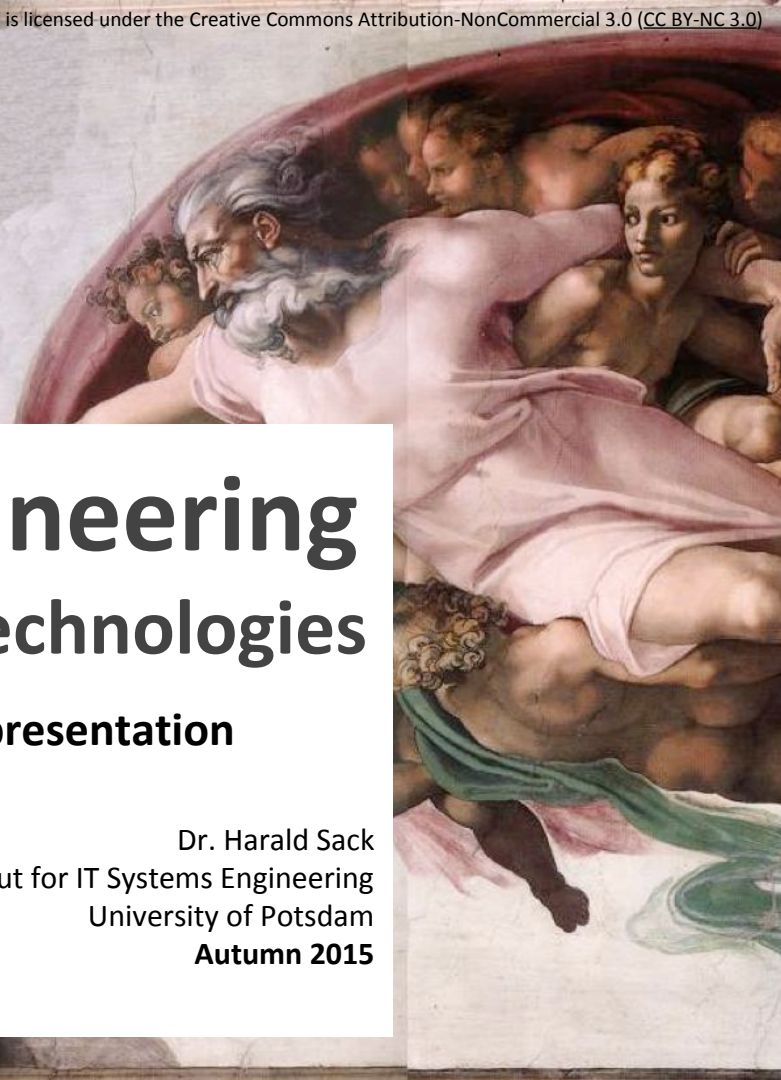


# Knowledge Engineering with Semantic Web Technologies

## Lecture 2: RDF Based Knowledge Representation 2.6 Complex Queries with SPARQL



Dr. Harald Sack  
Hasso-Plattner-Institut for IT Systems Engineering  
University of Potsdam  
Autumn 2015



# SPARQL - Filter Constraints

- Example: Filter results only for English labels

```
PREFIX : <http://dbpedia.org/resource/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT ?author_name ?title ?pages
FROM <http://dbpedia.org/>
WHERE {
    ?author rdf:type dbo:Writer .
    ?author rdfs:label ?author_name
    FILTER (LANG(?author_name)="en") .
    ?author dbo:notableWork ?work .
    ?work dbo:numberOfPages ?pages
    FILTER (?pages > 500) .
    ?work rdfs:label ?title .
    FILTER (LANG(?title)="en") .
} LIMIT 100
```

# More SPARQL Operators

- Logical connectives **&&** and **||** for xsd:boolean
- Comparison operators **=**, **!=**, **<**, **>**, **<=**, and **>=** for numeric datatypes, xsd:dateTime, xsd:string, and xsd:boolean
- Comparison operators **=** and **!=** for other datatypes
- Arithmetic operators **+**, **-**, **\***, and **/** for numeric datatypes
- and in addition:
  - **REGEX(String,Pattern)** or **REGEX(String,Pattern,Flags)**
  - **sameTERM(A,B)**
  - **langMATCHES(A,B)**

# SPARQL - Filter Constraints

- Example: Book titles that contain the word “love”

```
PREFIX : <http://dbpedia.org/resource/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT ?author_name ?title
FROM <http://dbpedia.org/>
WHERE {
    ?author rdf:type dbo:Writer .
    ?author rdfs:label ?author_name
    FILTER (LANG(?author_name)="en").
    ?author dbo:notableWork ?work .
    ?work rdfs:label ?title .
    FILTER (LANG(?title)="en")
    FILTER REGEX (?title, "love", "i").
} LIMIT 100
```

*string*

*regular expression*

*flags*

learn more about regular  
expressions at  
<http://regexone.com/>

[query SPARQL endpoint](#)

# SPARQL - Filter Constraint Evaluation

- SPARQL Filter Constraints are evaluated in 3-valued Logic
- truth values: `true`, `false`, and **error**

| A | B | A    B | A && B |
|---|---|--------|--------|
| T | T | T      | T      |
| T | F | T      | F      |
| F | T | T      | F      |
| F | F | F      | F      |
| T | E | T      | E      |
| E | T | T      | E      |
| F | E | E      | F      |
| E | F | E      | F      |
| E | E | E      | E      |

binary operators

| A | !A |
|---|----|
| T | F  |
| F | T  |
| E | E  |

unary operators

# SPARQL - Filter Constraints

- Example: Retrieve also the German book title, if available

```
PREFIX : <http://dbpedia.org/resource/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT ?author_name ?en_title ?de_title
FROM <http://dbpedia.org/>
WHERE {
    ?author rdf:type dbo:Writer .
    ?author rdfs:label ?author_name
    FILTER (LANG(?author_name)="en") .
    ?author dbo:notableWork ?work .
    ?work rdfs:label ?en_title .
    FILTER (LANG(?en_title)="en") .
    OPTIONAL {?work rdfs:label ?de_title
        FILTER (LANG(?de_title)="de") .
    }
} LIMIT 100
```

- The keyword **OPTIONAL** selects optional elements from the RDF graph
- complies to a Left Outer Join

*optional  
constraint*

# SPARQL - Alternative Results via UNION

- Example: Retrieve all influencers of and people influenced by Jules Verne

```
PREFIX : <http://dbpedia.org/resource/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
```

```
SELECT ?influencer ?influenced
FROM <http://dbpedia.org/>
WHERE {
  { :Jules_Verne dbo:influenced ?influenced . }
  UNION
  { :Jules_Verne dbo:influencedBy ?influencer . }
}
```

*logical  
disjunction*

- The keyword **UNION** allows for alternatives (logical disjunction)

# SPARQL - Negation

- Example: Retrieve authors that don't have an entry for "notable work"

```
PREFIX : <http://dbpedia.org/resource/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT ?author
FROM <http://dbpedia.org/>
WHERE {
    ?author rdf:type dbo:Writer .
    OPTIONAL {?author dbo:notableWork ?work . }
    FILTER (!BOUND(?work)) .
} LIMIT 100
```

*no variable  
binding*

- Negation in SPARQL
- complies to "NOT EXISTS" in SQL

# SPARQL - Negation (2)

- Example: Retrieve authors that don't have an entry for "notable work"

```
PREFIX : <http://dbpedia.org/resource/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT ?author
FROM <http://dbpedia.org/>
WHERE {
    ?author rdf:type dbo:Writer
    FILTER NOT EXISTS {?author dbo:notableWork ?work .}

} LIMIT 100
```

*filter query  
result for  
existency*

- SPARQL 1.1 also provides  
FILTER expressions NOT  
EXISTS and EXISTS

# SPARQL - Negation (3)

- Example: Retrieve authors that don't have an entry for "notable work"

```
PREFIX : <http://dbpedia.org/resource/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT ?author
FROM <http://dbpedia.org/>
WHERE {
    ?author rdf:type dbo:Writer
    MINUS {?author dbo:notableWork ?work .}

} LIMIT 100
```

*remove from  
query result*

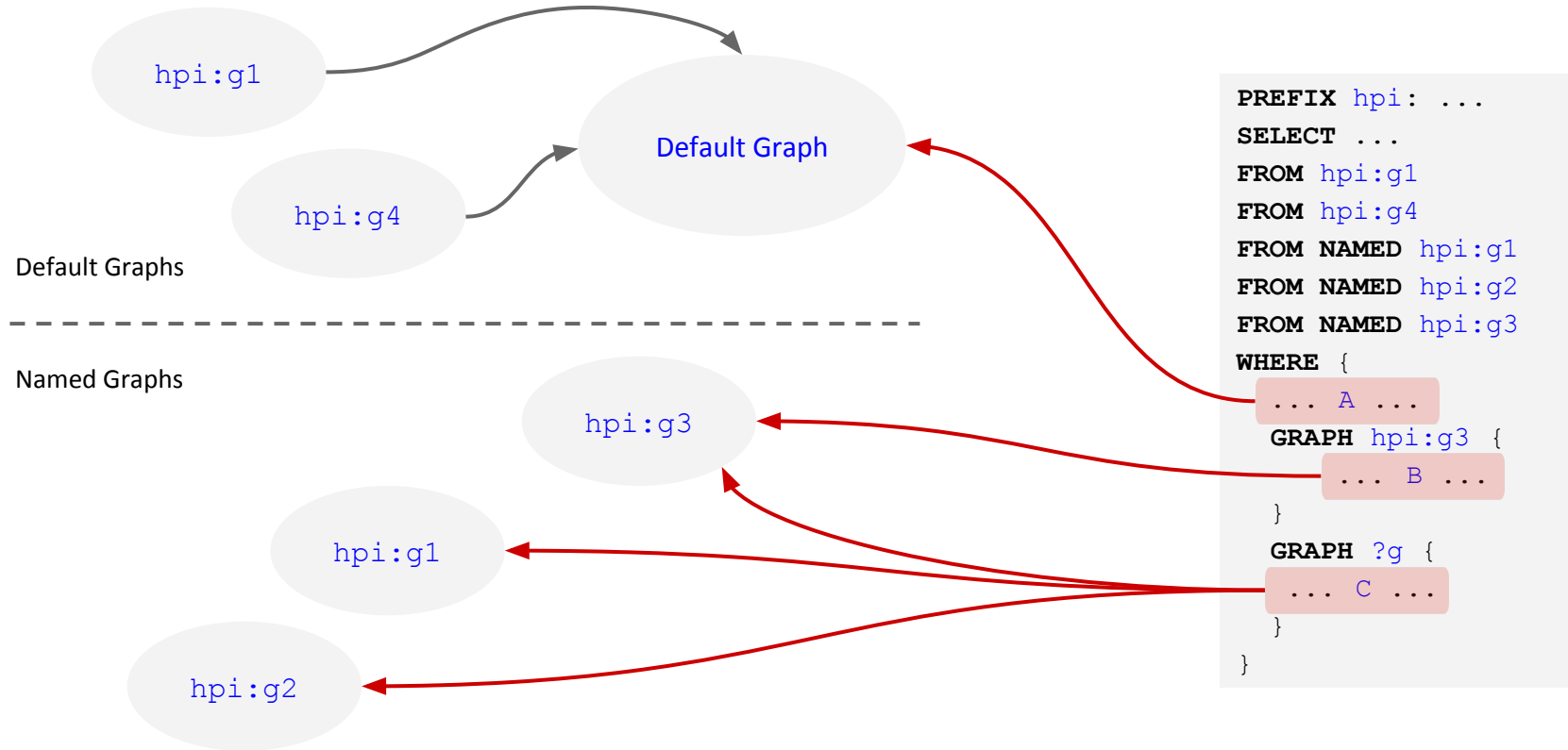
- Filtering of query solutions by removing possible solutions with MINUS.
- Difference to NOT EXIST:
- MINUS changes the graph pattern
- query result is dependent on position of MINUS

# SPARQL - RDF Graphs

- SPARQL queries are executed over an RDF dataset
  - one (or more) **default RDF graph** (FROM)
  - zero or more **named RDF graphs** (FROM NAMED, GRAPH)
- **Named Graphs** can be explicitly addressed via the keyword GRAPH and the URI of the named graph

```
SELECT ...  
WHERE {  
  ...  
  GRAPH <http://example.org/graph1.rdf> {  
    ?x foaf:mbox ?mbox  
  }  
  ...  
}
```

# SPARQL - RDF Graphs



# SPARQL - RDF Graphs

- How to ask a SPARQL Endpoint which RDF Graphs are available?

```
SELECT DISTINCT ?g
WHERE {
  GRAPH ?g { ?s ?p ?o . }
}
```

# SPARQL - Federated Queries

- SPARQL enables federated queries over several RDF datasets or SPARQL endpoints via the **SERVICE** objective

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX movie: <http://data.linkedmdb.org/resource/movie/>
PREFIX dcterms: <http://purl.org/dc/terms/>

SELECT ?film ?label ?subject WHERE {
  SERVICE <http://data.linkedmdb.org/sparql> {
    ?film a movie:film .
    ?film rdfs:label ?label .
    ?film owl:sameAs ?dbpediaLink
    FILTER regex(STR(?dbpediaLink), "dbpedia", "i")
  }
  SERVICE <http://dbpedia.org/sparql> {
    ?dbpediaLink dcterms:subject ?subject .
  }
}

LIMIT 100
```

- Example: connect the **Linked Movie Database** with **DBpedia**
- only possible, if SPARQL endpoints permit federation

# SPARQL



**Next: 07 - More complex SPARQL queries**

OpenHPI - Course Knowledge Engineering with Semantic Web Technologies

**Lecture 2: RDF Based Knowledge Representation**